PhUSE 2011

Paper PP02

Define your Own SAS® Command Line Commands

Duong Tran, Independent Contractor, London, UK

ABSTRACT

SAS® provides a number of Command Line commands and some of these are frequently used. For example the commands **fsv**, **lib, var** etc. But did you know that you can create your own? This paper will show how you can customize repetitive actions of a particular task into a Command Line command.

Keywords: gsubmit, cmdmac

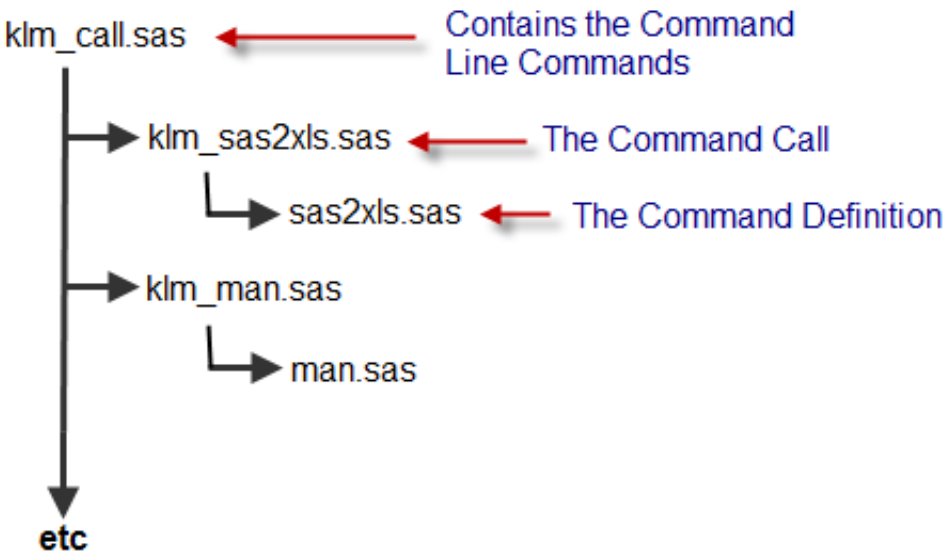
INTRODUCTION

SAS allows you to customize a sequence of actions into a simple command which is very useful. For examples often you want to know – what are the variables in the dataset, the definition for a format etc. You could define a macro for each of these then assign it to function keys, but a major drawback with this technique is that you will soon run out of function keys. A better way is to design a command that you can execute at the command line, just like those that are already existed in SAS. The SAS **gsubmit** command and the **cmdmac** option make this extension possible.

DEFINING SAS COMMMAND LINE COMMANDS

The desire is to define commands in the following form; **COMMAND** <*parameter(n)*> | '<*parameters a string*>'>. That is a command should be able to take parameters (options) and therefore make its versatile. Similarly to the **ls** command in UNIX for example, where it can takes options **-la** etc.

Here is a schematic of the design for defining Command Line commands follow by a complete practical example. Note the advantage of this design is that your command macro is just a "normal" macro like any others, that is to say you can turn any macros you already have into a Command Line command if so wish.



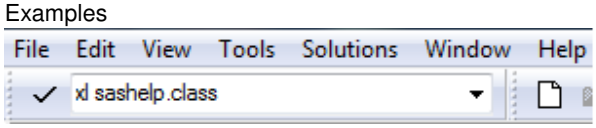
Example

This is a complete example to send the specified dataset to an Excel file. The **fundamental** is the **cmdmac** and **cmd** options that allow parameters entered at the Command Line to be taken into a macro as macro variables. See `(dset, vars, ofile)` below.

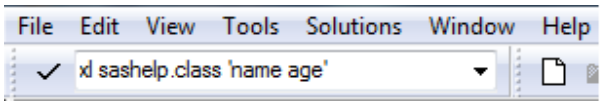
```
*klm_call.sas;
options cmdmac; * must have option;

* send data to excel;
%macro xl(dset, vars, ofile) / cmd;
  gsubmit "%inc 'klm_sas2xl.sas';";
  output;
%mend;
*more commands etc..


*klm_sas2xl.sas;
%macro sas2xl_klm;
 %inc "sas2xl.sas";
 %sas2xl(dset  = &dset
        ,vars  = %sysfunc(compress(&vars, %str(%"%')))
        ,ofile = &ofile
        );
%mend;
%sas2xl_klm;

*sas2xl.sas;
%macro sas2xl(dset  =
             ,vars  =
             ,ofile =
             );

  %local vlist lword l1 l2;
  %let vlist = &vars;
  %let lword = %scan(&vlist, -1);

  * if option specified;
  %if %datatyp(&lword) = NUMERIC %then %do;
    %let l1 = %length(&vlist);
    %let l2 = %length(%scan(&vlist, -1));
    %let vlist = %substr(&vlist, 1, %eval(&l1 - &l2));
  %end;

  * delete the view initially;
  proc datasets nolist library=work mt=view;
    delete dsview;
  quit;

  * create dataset view;
  data dsview / view=dsview;
    retain &vlist;
    %if &lword = 1 %then %do;
      set &dset(keep=&vlist);
    %end;
    %else %do;
      set &dset;
    %end;
  run;

  * if output file is not specified then default it to this location;
  %if &ofile eq %then %do;
          %if %upcase(%scan(&SYSSCP,1))=SUN %then %let ofile = %sysget(HOME)/temp.xls;
    %else %if %upcase(%scan(&SYSSCP,1))=WIN %then %let ofile = c:\temp.xls;
  %end;

  %put output file: &ofile;

  * export view to excel;
  proc export
    data=dsview
    outfile="%scan(&ofile, 1, %str(.))"
    dbms=xls
    replace;
  run;

%mend
```

You can execute this Command Line as:

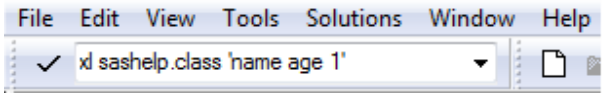
Examples



export dataset to Excel



as above but name and age in col1 and col2



as above but keep name and age only

Note: string in quotes will be treated as a single parameter. This possibility is useful because you can pass a string as a **single** parameter and then parse it inside the command call macro.

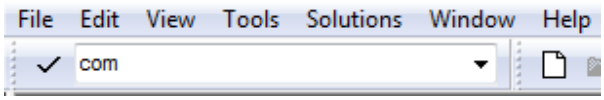
DISPLAY THE RESULT OF THE COMMAND

The above example send the result to a file but there will be commands where you would prefer to send the result to the screen. And here are a number of ways to do this.

1. Send to the **output** panel with Proc Print, Report etc
2. Send the result to a text file and use Proc Fslist, Notepad or Nedit etc to display this text file
3. Send the result to a HTML file via ODS and use a Web Browser display this HTML file

SOME IDEAS FOR MORE COMMANDS

How about a command **com** to list the available commands?



Command Line Commands

1. **man** - to display available commands
2. **man** <command> - to display the command documentation
3. **fmt** <fmtname> - to display the definition of the format

etc .. the list is up to your imagination

CONCLUSION

The technique shown in the paper allows you to define 'any' Command Line command imaginable. It's simple yet powerful. That is a Command Line command is just a macro like any other macro. Also note the SAS x statement can call other applications and therefore a command could be written in other languages for example PERL.

REFERENCES

[1] Tran, Duong "Improving Programming Efficiency for Statisticians and Programmers with Customization" – PSI 2010
[2] Benjamin, Neely "COMMAND Your Session: SAS® Commands and Command-Style Macros – SAS Global Forum 211
[3] DeVenezia ,Richard A. "SAS® Explorer: Use and Customization" – NESUG 18

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Duong Tran
Work Phone: 07752326413
Email: trand000@aol.com
Web: www.tranz.co.uk